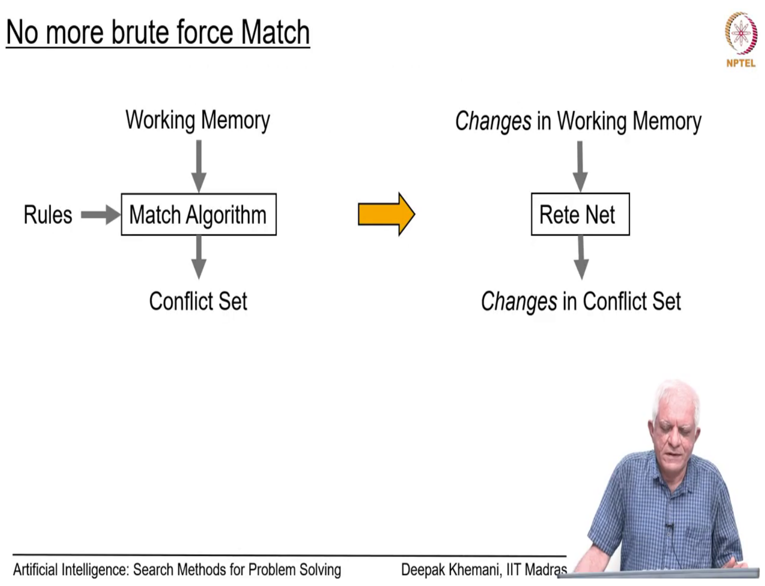
Artificial Intelligence: Search Methods for Problem Solving

Prof. Deepak Khemani
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Chapter – 06
A First Course in Artificial Intelligence
Lecture – 77
Rule Based Expert Systems
The Rete Net

(Refer Slide Time: 00:14)



No more brute force Match

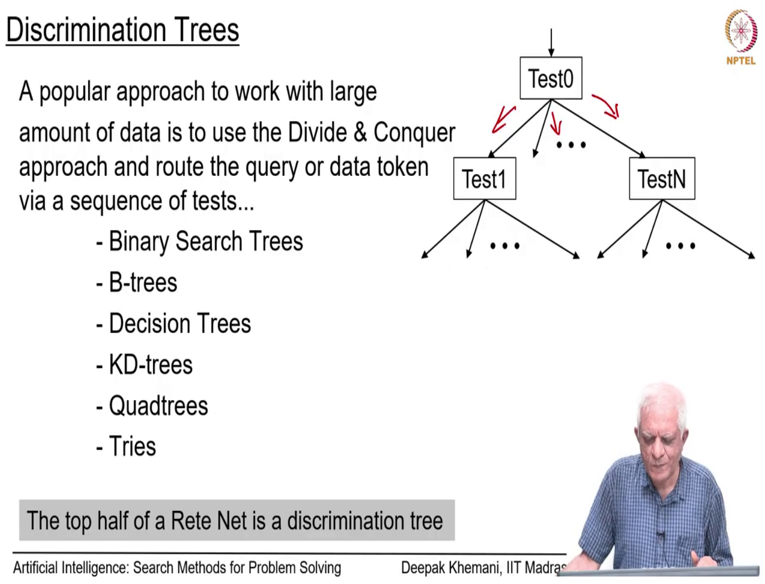
$$\begin{tikzcd} & \text{Working Memory} \arrow[d] & & \textit{Changes}\text{ in Working Memory} \arrow[d] \\ \text{Rules} \arrow[r] & \boxed{\text{Match Algorithm}} \arrow[d] \arrow[rr, Rightarrow] & & \boxed{\text{Rete Net}} \arrow[d] \\ & \text{Conflict Set} & & \textit{Changes}\text{ in Conflict Set} \end{tikzcd}$$

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

We are just about to look at this very nice algorithm called the Rete algorithm which works on something called the Rete net. And, the goal of constructing this Rete net as we have mentioned is to optimize on the work that we have to do during match phase of the cycle of the infra engine cycle match resolve execute in every cycle we want to do the minimum want of work of match.

And, even within a cycle we want to share the match needed by different rules and different patterns if it can be shared and done only once we would like to do that and the Rete algorithm does that very nicely or the Rete net increments that very nicely.

(Refer Slide Time: 01:10)


Discrimination Trees

A popular approach to work with large amount of data is to use the Divide & Conquer approach and route the query or data token via a sequence of tests...

- Binary Search Trees
- B-trees
- Decision Trees
- KD-trees
- Quadtrees
- Tries

The top half of a Rete Net is a discrimination tree

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, let us get on with this. The basic idea that we want to start with is a general concept used a lot in computer science. These are discrimination trees and the idea of discrimination trees is to organize data in such a way that finding specific species of data becomes faster essentially.

Where is a notion of finding for us? Essentially we are going to say that we have a new working memory element remember that we are going to be making only changes into the working memory element and as a output we would expect changes into the conflict set.

So, assume that we have this process going to be more precise if you look at this algorithm look at this diagram again say the changes in the working memory result in changing in the conflict set. How does one start? At the very beginning the working memory is empty to start with you start with application which means you load the Rete net and the initial working memory you put it at the top as we will see put it down the network.

So, it will serve as changes because the first time that you are adding the entire data it is changing the data, but subsequently all the data that you have entered earlier will remain inside the system hence only the new data you create will be added to the system.

So, this new data that you add to the system which is the Rete network needs to find go and find which rules it matches ok. So, this process of the working memory or data looking for a matching rule is what we are going to try and do this here and we will use this very common idea used in many areas of computer science which is called discrimination trees essentially.

The basic idea of discrimination trees is to keep distributing the patterns which need to be matched, so that instead of sequentially trying out all patterns once you have tested for one let us call it attribute of a pattern, then you have to only look at a smaller segment of the target set essentially.

In our case the target set is patterns in the rules but you can think of it as the other way around also that there is a pattern in the rule which is looking for of matching piece of data. So, all patterns with all data we are just trying to change into that.

This if you take the analogy from the blood vessel network that we were talking about it is like saying that you know from the heart you send out blood and it goes all over the body looking for various organs to replenish with oxygen essentially anyway.

So, in computer science in computing, a popular approach to work with large amounts of data is to use a divide and conquer approach and route the data or route the query rather or data token via a sequence of test.

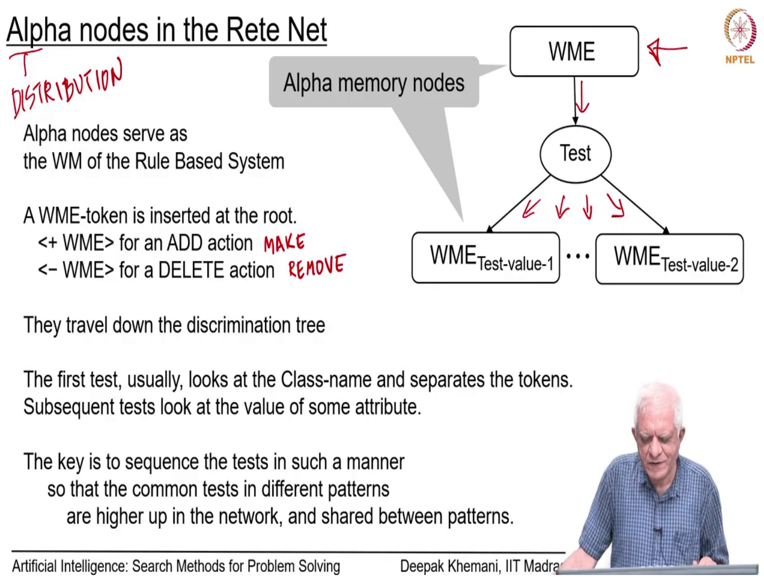
So, the generic diagram of such a discrimination tree looks like this that you do a test which we call as Test0. Then based on the value of the Test0 if it will go down to one of these branches below that either here or here or here or somewhere and then it will do another test and again it will get distributed and so on.

So, I am sure you are familiar with this kind of trees in a most common of them are binary search trees with all students start with essentially when they look at data structures.

So, binary search trees and B-trees and decision trees if you have looked at machine learning KD trees, K-dimensional trees which as used in this area of K space listening quite a bit, quadtrees which have to do with images, tries which have to do with words and alphabets and, how to you know match strings.

There are various kinds of things all of them rely on this basic idea of discrimination which says that you do a test and send the query or the data down a particular branch and then do another test and so on. So, the top half of our Rete net is going to be a discrimination tree.

(Refer Slide Time: 05:59)


Alpha nodes in the Rete Net

DISTRIBUTION

Alpha nodes serve as the WM of the Rule Based System

A WME-token is inserted at the root.
<+ WME> for an ADD action MAKE
<- WME> for a DELETE action REMOVE

They travel down the discrimination tree

The first test, usually, looks at the Class-name and separates the tokens.
Subsequent tests look at the value of some attribute.

The key is to sequence the tests in such a manner
so that the common tests in different patterns
are higher up in the network, and shared between patterns.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

Alpha memory nodes
WME
Test
$WME_{Test\text{-}value\text{-}1}$
$WME_{Test\text{-}value\text{-}2}$
NPTEL

This is how it will look. We have a working memory sitting in some location in the network and, the top half will be made up of what is called as alpha memory nodes. Alpha memory is will be concerned with distribution. It will spread the tokens along different directions and how will that happen?

The working memory comes down goes and does the test and based on a value of the test it goes down one of these parts. This one or this one or this one or this one as the case may be, and it will go and sit in one of these alpha nodes. So, these alpha nodes will actually store the working memory elements.

So, the network itself will serve as a working memory essentially. So, I said a little while ago that the network is a compilation of the rules, but this structure that we are building also stores as the memory for storing the data essentially. Of course, the memory is dynamic it is a short term memory that we talked about that tokens come and go and so on. Tokens by tokens you mean working memory elements they come and go and so on, but the network remains the same.

So, the network is like a long term memory of a problem solver and the working memory is the short term memory yes. So, what is the problem that you are solving? What is the data that you are addressing and things like that.

So, the alpha network the top half of the Rete network is a distribution network and which basically discriminates on the working memory on some attribute and sends it down some path essentially. So, when you implement this, it will essentially basically mean that you take a working memory elements and do a test on it and then send it to some location.

But, very often for the purposes of illustration we prefer slightly more compact diagrams and if you look at literature they are two different kinds of diagrams that you can think of yeah. So, as I said alpha nodes serve as a working memory for the rule basis. So, everything is inside the network a memory as well as the rules essentially.

So, our working memory element is inserted at the root. It is inserted as a token. So, it is either a positive token or a negative token; positive token if you have an ADD action in the rule that you are executing or the MAKE action if you remember sometimes we call the MAKE offs why calls it MAKE some other textbooks call it ADD. So, offs why uses the term MAKE and REMOVE; some books use ADD and DELETE, but, but for with ADD or MAKE we have a positive token and with DELETE we have a negative token.

So, essentially if you remember the rules it says delete this token, delete this working memory element and then it has to go back and somehow delete it. Now, conceptually of course, what we will do is that instead of as a process of finding that token as to where is it lying in the memory we will again go through this process of search that this discrimination tree does and the only difference is it is a working memory element which will go through the test which are there in the discrimination tree.

The prefix of plus or minus says whether you are adding it or whether you are going to delete it. So, if you have a; if you have a delete action then we will put down a negative token until go and find the positive token that existed earlier and it will delete it essentially. So, the tokens whether they are a positive tokens or negative tokens they travel down the discrimination tree and find some location essentially.

The first test in a Rete network, typically it looks for the class names. So, we have different class names right. We said that rank, next rank is so much or student name is so much or scores is a so much or whatever the class name may be we first separate the tokens for different classes and then of course, look at the attributes and so on.

From the implementation perspective, the key is to figure out what is the order in which you will do the test. So, I have said earlier that into the working memory as well as in the patterns in the rules the order of attributes does not matter. So, it is not that the attribute that you mention first is the first one that should be tested and so on.

Of course that will be the most straight forward we have doing things, but in practice if you can figure out as to in what order to test for example, which attribute to test next might allow you to share more of the work than if you were to do the same test distributed in different branches essentially. So, if the same test is being done in this one branch here, one branch here and one branch there, you can move the test up in the hierarchy so that it is done only once essentially.

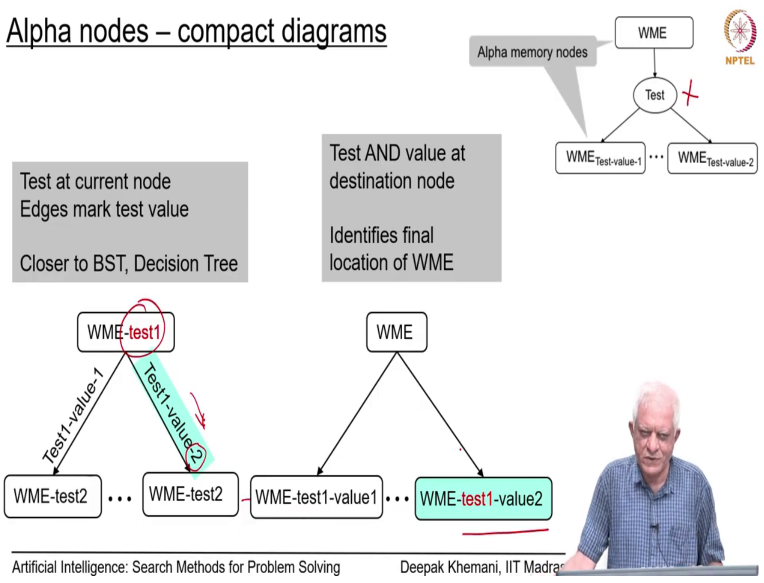
So, that is going to be a goal in building an efficient Rete network. So, those are few have studied decision trees would remember that an algorithm called ID3 uses this notion of information gain to decide as to what is the next test that should be do done and the test should be such that it separates the data as much as possible essentially.

And, the same thing something similar we want to do here except that here we want to share the work that is being done by different patterns essentially. We will not go into the details of this implementation and hope you will think a little bit about how to do this essentially. So, the key is to sequence a test which means which attribute should be test higher up in the tree and which attribute should be test tested lower down in the tree.

And, the attribute that you want to test higher up is the one which is being used in many different patterns essentially. It may be the same class leap class this thing, but you may be testing for different things.

So, student name, roll number, marks and so on ok. So, as I was saying this is what you would do if you were implementing the system. You would take a working memory element feed it to a test. And, the test would directed into different directions based on where it is going, but for purposes of illustration we often use more compact diagrams.

(Refer Slide Time: 13:13)



Alpha nodes – compact diagrams

Test at current node
Edges mark test value

Closer to BST, Decision Tree

Test AND value at destination node

Identifies final location of WME

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

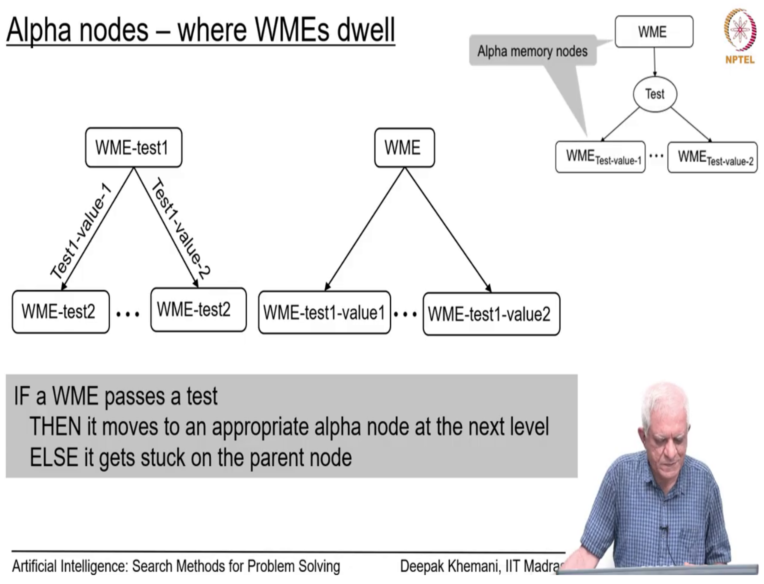
And, we essentially eliminate drawing the test node as a separate node and we just work with a alpha nodes that are there. And, we either draw a diagram which looks like this where the test is done at the current node. So, we can see at the top here that the test is being mentioned here and the value of the test is mentioned on the edge that follows essentially.

This is only for illustration purposes. If they are easier to draw and use when we are talking about Rete net and supposing the test value 2 is the one that is being matched by a particular working memory element the test is Test1 and the value is 2 and so, the in the node with the test value 2 will come here and sit here and here of course, it will do another test. Here it will do another test which is test2 which is the next test essentially.

So, as you can see we have eliminated the test node and just simply remember the working memory nodes essentially which are the alpha nodes. An alternative way of representing the same effect is to mention the test at the destination node or at the target node. So, because the test1 had value 2 we have kind of mention this whole thing here in one place and in some sense it kind of pulls relevant token from its parent node essentially.

So, the basic idea is that the input token was at the parent node and it has to go to one of the children and that which will which child it goes to is determined by the test that we are doing on that node essentially.

(Refer Slide Time: 15:01)



Alpha nodes – where WMEs dwell

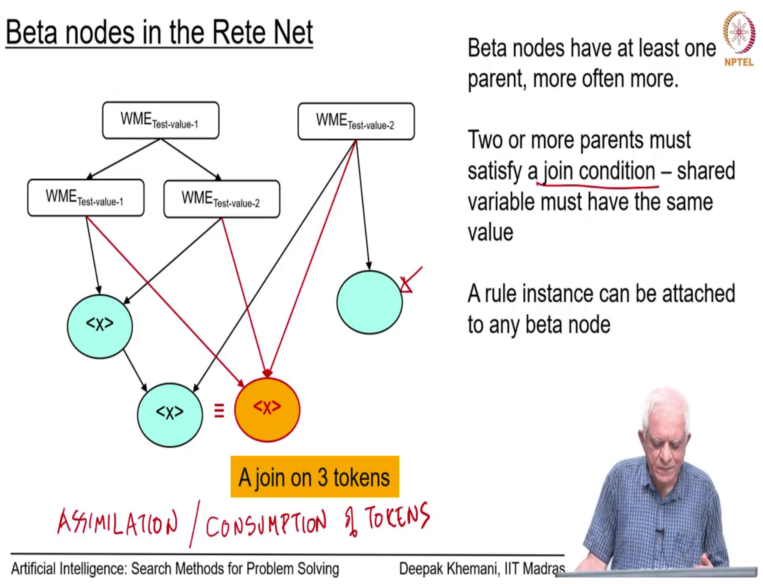
IF a WME passes a test
THEN it moves to an appropriate alpha node at the next level
ELSE it gets stuck on the parent node

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, these are the two diagrams that we have seen. In if the working memory element passes a test whichever test we are doing, THEN it moves to the appropriate alpha node in the next level ELSE it get stuck on the parent node. So, you can imagine now we had this network we put the token from the top a working memory element and it traverses down the network essentially. As long as it passes some of the test which have being tested for in the network essentially.

Again, remember that they can be both positive nodes as well as negative nodes as far as finding their destination they are whether they are positive or negative does not matter. What the semantics of positive and negative is that positive means that is a new piece of data is being added to the working memory, negative means that in existing piece of data is being deleted from the working memory, but the location is specified by the alpha network essentially.

(Refer Slide Time: 16:05)


Beta nodes in the Rete Net

Beta nodes have at least one parent, more often more.

Two or more parents must satisfy a join condition – shared variable must have the same value

A rule instance can be attached to any beta node

A join on 3 tokens

ASSIMILATION / CONSUMPTION of TOKENS

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

NPTEL
$WME_{Test-value-1}$
$WME_{Test-value-2}$
$WME_{Test-value-1}$
$WME_{Test-value-2}$
<x>
<x>
≡
<x>

So, the lower part of the network is when assimilation network or a consumption network. Assimilation in the sense that a rule will has many patterns it needs one working memory element for every pattern in the rule and it is this lower part of the network which is called the beta network or beta nodes in the network, they assimilate these different tokens which the rules required essentially.

So, in some sense they are consuming, but you can consume tokens only if all the patterns in the rule match and this part of the match process is done by the beta network essentially. So, beta networks have at least one parent node more often they have more in this diagram that I have drawn; I have drawn three beta nodes. And, one of them has only one parent the one on the right this one here and the other two have two parents each essentially.

If a beta node has two parents, then it must satisfy the join condition and the way that we do join in data basis as well is that if we have a shared variable then the variable in the two tokens that are coming in must have the same value essentially. So, remember that we had said that if there is a student name S and A student name S has got.

So, many marks then that S and S must match the same student; whether it is Sneha or Sunil or Anil or Shruthi whatever the case may be the variable which is in mentioned in the pattern of the rules remember variables are only in rules, the working memory elements themselves will have only data only constants.

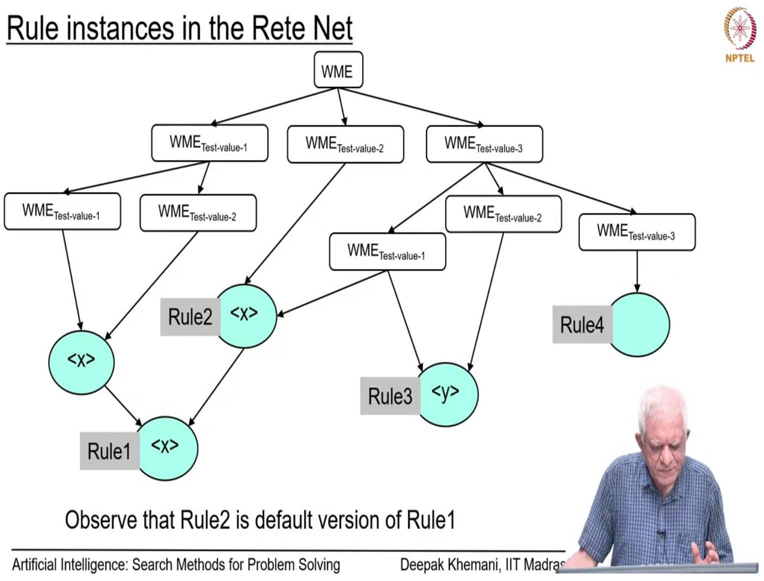
So, the variable that particular rule is talking about let us say in this example we have called it x in the two pieces of data which are flowing down must have the same value. So, that obviously, is as we say a the join condition. If the join condition holds, then the two patterns can come together and hopefully contribute to the rule becoming ready to join the conflict set.

Now, rules can be attached to any beta nodes essentially ok. So, that is a difference that is one difference between beta node and alpha nodes in this diagram that we have drawn that if a token or a set of token reaches some beta node, and if that beta node has sufficient number of tokens to satisfy a rule then you can say that the rule is matching essentially. In this way simple diagram on the beta node on the right side seems to have only one condition.

So, there is a very simple rule. Remember that pass rule that I had mentioned about in bridge building such rules may be something like this essentially. Now, I have drawn here two beta nodes on the left and one of them is the parent of the other beta node. I could easily have written drawn this network a little bit differently where the join is happened on three tokens there is no difference in the meaning of the semantics of whichever way I construct the network.

The basic idea is that there are three tokens which have to come together either they can either first two of them can come together and then they can join the third token or all three can come together at the same time as shown in that new diagram. The key thing is that the variable in those arcs these variables are associated with this paths in the network essentially, must have the same value in the tokens which are coming in essentially then we say that they have joined.

(Refer Slide Time: 20:57)



Rule instances in the Rete Net

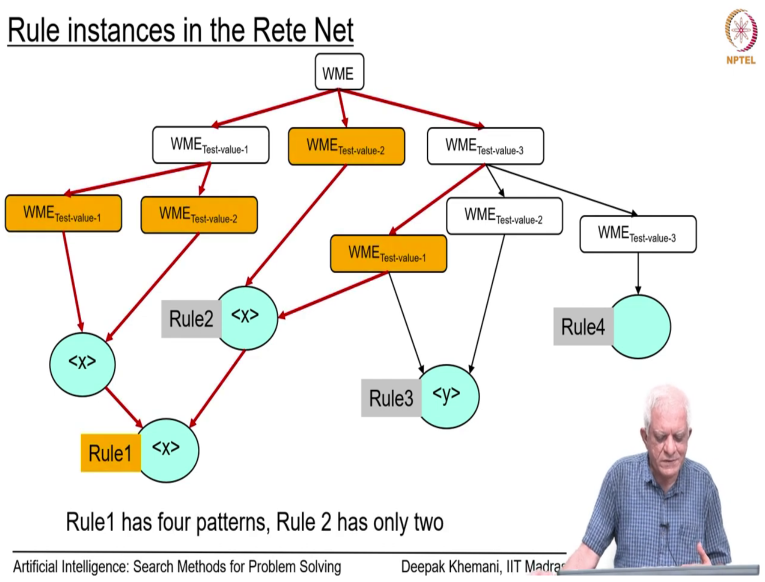
WME
WME$_{Test\text{-}value\text{-}1}$
WME$_{Test\text{-}value\text{-}2}$
WME$_{Test\text{-}value\text{-}3}$
WME$_{Test\text{-}value\text{-}1}$
WME$_{Test\text{-}value\text{-}2}$
WME$_{Test\text{-}value\text{-}2}$
WME$_{Test\text{-}value\text{-}3}$
WME$_{Test\text{-}value\text{-}1}$
Rule2 <x>
<x>
Rule1 <x>
Rule3 <y>
Rule4

Observe that Rule2 is default version of Rule1

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, here is an example of a Rete network which has a some number of rules. So, it is a network for some four rules and as a I said a network is a compilation of rules. So, if you had written those four rules we have not said what those rules are, but you can imagine that you have written some four rules which resulted in this particular network being formed here essentially. And, again you can see that Rule2 is a parent of Rule1 and this means that Rule2 is a default version of Rule1. Why do we say that?

(Refer Slide Time: 21:46)


Rule instances in the Rete Net

Rule1 has four patterns, Rule 2 has only two

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

If you look at these two rules you see that rule two needs only two patterns to match which are coming in along. So, so this red edges are the edges along which the patterns which match this Rule1 travel from.

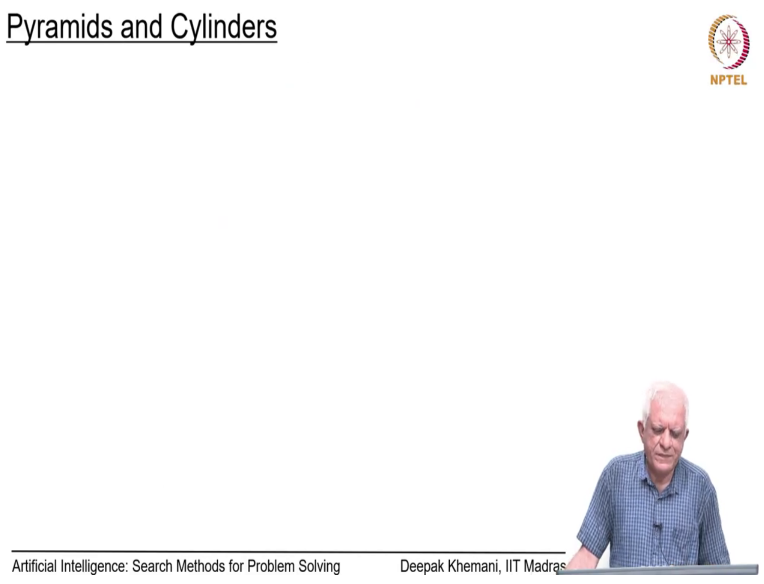
So, we can see that there are four paths from the root to Rule1 and each of those four paths represents one pattern in a rule. So, this Rule1 clearly has four patterns in the left hand side it needs four tokens to come and match those four tokens the four tokens will travel along the four paths that we have shown here in red here.

So, if all the four tokens reaches Rule1 ah, then we will say that Rule1 is matching and we will put it in the conflict set with the four tokens that have come there. Rule2 takes only two of those four tokens. The same four tokens that we talked about for Rule1, Rule2 was happy with only two of them.

And, so, clearly you can see that it is a default version of the Rule2. If you had all the four tokens you would say Rule1 is more specific and I would like to use Rule1 if you had only two or three of those four tokens and Rule2 was matching then Rule2 would still be applicable.

So, Rule2 is kind of a default version of Rule1. Again, I will draw remind you of two rules we talked about in bidding – one was to pass and the other one was to bid one more term ok. So, that is the idea of beta networks and alpha networks and this whole thing is called the Rete network essentially.

(Refer Slide Time: 23:24)



Pyramids and Cylinders

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, let me stop here and take a break. And, we will look at some examples in the next session or next video which we will talk about some specific set of rules and how they are combined into a Rete network ok.

So, we will see you in the next session.